Image processing tasks applied to robot vision system and path discovery

Italo Guedes A. Silva, Rafael C. Cejas. Advisors: Prof. Sarosh Patel, Prof. Tarek Sobh
Interdisciplinary RISC Lab
School of Engineering, University of Bridgeport

Abstract

This work presents the project of a mobile robot with an attached handler, programmed to collect small objects in the near area. The robot takes photos –is able to process a constant stream of images– of the surround area searching for objects, and goes to the nearest one to collect. After collecting, the robot has to put the object in a specified place. The robot will keep searching for objects and moving while on. The image processing job is to do a contrast between the floor color and the colors of the objects, and then the algorithm provides the location to move on. The project is built with a Raspberry Pi/Raspbian platform running a python program that uses the OpenCV library to process images and parameters used to guide the robot, a simple webcam that provides high resolution images, and the motors as final actuators to control the movement of the robot and the handler.

Image Processing

The application in this project uses the camera features of the OpenCV library to obtain a constant stream of images from the webcam. Every frame is processed with transformations to eliminate any noise that can make harder to distinguish the objects.

Figure 1: Original frame converted to grayscale, and to HSV color system.

cvtColor - This function is used to convert the original image to other scale of colors. This is an important task to do before applying the filters. In this project, we used this function to convert the captured image to grayscale for the objects detection, and to HSV to distinguish the floor and the walls. In Figure 1, you can see the input image and the converted ones.

Gaussian Filter - The first step in processing the image to detect objects is to smooth or blur the image. This effect has the main objective of decrease the small details and the high-frequency noise in an image. The function is called *GaussianBlur*.

Figure 2: Gaussian filter

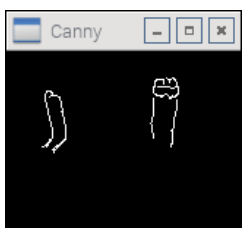

Figure 3: Canny edge detection

Canny edge detection - This function identifies the edges of an image by detecting regions with rapid color intensity variation. At this time probably all edges were identified, but if there is any pixel left that are not part of an edge still on the image, they will be eliminated, and will select which pixels should remain in the image and which should be deleted based on value of the pixel gradient (P), using the lower threshold (T_{lower}) and the upper threshold (T_{upper}), as you can see in figure 4.

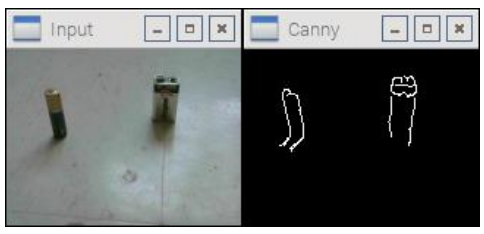

Figure 4: Canny edge detection

Figure 5: Result of MorphologyEx

MorphologyEx – In this function, the image goes through an erosion effect, removing small objects and possible artifacts, followed by a dilation effect, which dilates the remaining pixels to enhance the size of the real objects in the image. The result you can see in the Figure 5.

Path Discovery

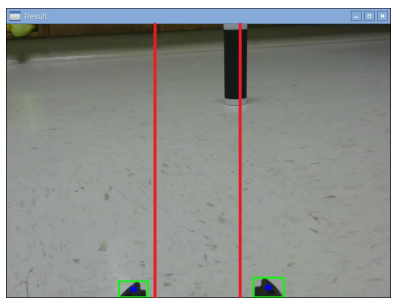

Figure 6: Object centralization

To deal with object detection, we use a function called findContours to obtain details of the objects in the frame. The cv2.findContours uses a binary image as input and returns a set of contours found in the frame. The movements –to left or to right– are executed by the robot in order to change the camera's position to a frame that contains the object located in the middle of the screen (Fig. 6).

Results

The workspace consists of a square area surrounded by a line to identify the boundaries, as can be seen in Fig. 7. The objects in cylindrical form were chosen so the robot's gripper can grab them easily from every possible position.

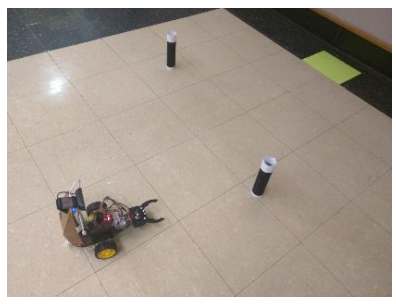

Figure 7: Robot moving to target

Figure 8: Robot leaving object in marked area.

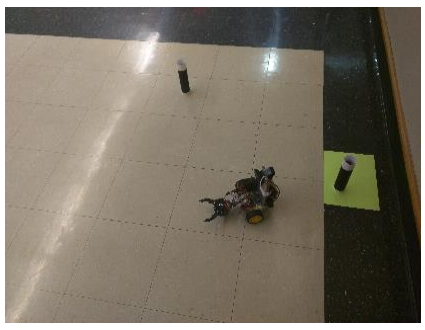

Figure 9: Robot returning to workspace.

Fig. 8 shows a snapshot of the moment the object is dropped. Fig. 9 shows the robot returning to the workspace.

Conclusion

In the presented work we could learn about some of the powerful functions of OpenCV for image processing, and how they can be used with basic techniques to extract what we want from the workspace. With this project we could begin our journey into the image processing world, becoming more interested in the relationship that it has with robotics, electronics and programming.

References

1. J. R. Parker, *Algorithms for Image Processing and Computer Vision, 2*th ed. Indianapolis, IN. Wiley Publishing Inc., 2010, pp. *1–177.*
2. OpenCV Documentation [Online]. Available site: http://docs.opencv.org/2.4/doc/tutorials/imgproc/table_of_content_imgproc/table_of_content_imgproc.html